Facility Power Usage Prediction with Artificial Neural Networks

A Thesis

Presented to the Faculty of

California Polytechnic State University

San Luis Obispo, California

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science in Electrical Engineering

by

Sunny Wan

June 2009

COMMITTEE MEMBERSHIP

TITLE: Facility Power Usage Prediction with Artificial Neural Networks

AUTHOR: Sunny Wan

DATE SUBMITTED: June 2009

COMMITTEE CHAIR: Dr. Helen Yu, Associate Professor

COMMITTEE MEMBER: Dr. Taufik, Associate Professor

COMMITTEE MEMBER: Dr. Wayne Pilkington, Assistant Professor

ABSTRACT

Facility Power Usage Prediction with Artificial Neural Networks

by Sunny Wan

Residential and commercial buildings accounted for about 68% of the total U.S. electricity consumption in 2002. Improving the energy efficiency of buildings can save energy, reduce cost, and protect the global environment.

In this research, artificial neural network is employed to model and predict the facility power usage of campus buildings. The prediction is based on the building and the weather conditions such as temperature, humidity, wind speed, etc. Various neural network configurations are discussed; satisfactory computer simulation results are obtained and presented.

ACKNOWLEDGMENTS

First, I would like to express sincere appreciation to Dr. Helen Yu for her assistance in my research and the preparation of this manuscript. It is a great challenge for me to do research in artificial neural networks. Dr. Helen Yu's vast knowledge in artificial neural networks guided me to overcome many difficulties in my research.

In addition, special thanks to Dennis Elliot for providing me the power consumptions data that was used in this research, answering my questions about the data, and opening my mind to the world of energy management.

Thanks to Dr. Taufik for helping me out on issues about power. Thanks to Dr. Pilkington for keeping track of the progress of my thesis, and communicating with the International Student Program Office about the status of my student VISA.

Last but not least, I need to thank my parents for supporting my living in U.S. I would not be able to complete my education without their support and encouragement.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

Chapter 1: Introduction

1.1 Purpose of this research

With the limited resources of fossil fuel and the ever – increasing demand of energy, the studies on power efficiency have become more and more important.

Prediction plays an important role in power optimization. In this research, artificial neural network is employed to model and predict the facility power usage of Cal Poly campus building. Computer simulation results show that the adaptive line ability and nonlinear mapping ability of neural network fit well with the problem and thus artificial neural network provides an alternative approach for power prediction.

1.2 Traditional method

1.2.1 The Simulation Modeling Method

Preliminary Energy Assessments [13] is a simple analysis method based on information gathered by surveys and measurements. It can be used to predict yearly building electricity consumption based on building schedule and seasonal changes. Energy Feasibility Studies offer more detailed analysis by modeling the physical structure of the building and the facility usage, such as building envelope, building occupancy rate, lighting, HVAC equipment and ventilation rate. In practice, the exact measurements for variables in the simulation models may not be available; thus the model may not match with the real-world situation.

1.2.2 The Regression Modeling Method

An approach frequently used to determinate the power consumption of commercial buildings is regression modeling [11]. Since the electrical power consumption of a building is a function of several variables, multiple linear regression models would be more appropriate for electrical energy consumption prediction than a simple variable.

Linear regression methods are widely used to approximate an input-output mapping via measurement. It is easy to implement; however, it may not be very accurate when comparing with other analysis methods.

1.3 Why Artificial Neural Networks?

The approach based on artificial neural networks is a non-parametric technique that can approximate complex functions and is promising for many applications

When compared with traditional methods for building power consumption prediction, artificial neural networks are more reliable, provide faster learning time with increased simplicity in analysis and adaptability to variations in weather conditions and building's energy usage.

Once an artificial neural network model is fully trained, engineers can easily use model to predict and evaluate the building energy efficiency performance without needing detailed knowledge of the artificial neural networks method.

Artificial neural networks learn the input/output mapping of a system through an iterative training process. They can also update their learned knowledge on-line over time. This automatic learning property makes a neural network based system inherently adaptive.

The prediction from artificial neural network model can be used to optimize the power distribution in a building, and thus spread the demand of power over the day, reducing maximum demand charges.

Chapter 2: Artificial Neural Networks and its Applications in Building Energy Consumption Prediction

2.1 Overview of Artificial Neural Networks

The computational structure of a multi-layer feed-forward artificial neural network consists of an input layer, which accepts patterns from the environment; an output layer which shows network response to the environment; and one or more hidden layers that connect the input layer and output layer together; however, they do not interact directly with the environment.

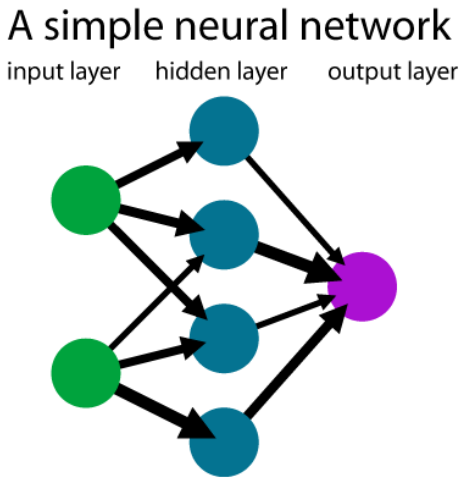

Figure 2-1: A multi-layer feed-forward neural networks [1]

In the neural network training process, weights are modified. Defined relations between the input layers, the hidden layers and the output layers determine a particular neural network model. Three types of networks used most commonly in artificial neural networks applications are feed-forward networks, competitive networks and recurrent

associative memory networks. Furthermore, each network type may have different learning rules. The learning rules are described in broad categories of supervised learning, unsupervised learning and reinforcement learning rules.

2.2 Review of Artificial Neural Networks in Energy Prediction

The artificial neural networks have been investigated for their applicability in building energy predictions over the past ten years ([2] [3] [4] [5] [6]). Various neural network architectures have been applied in energy predictions. They include back-propagation, recurrent neural networks, autoassociative neural networks and general regression neural networks and have demonstrated relatively successful results having coefficient of variations in the range of 2–40% ([2] [3] [4] [5] [6]). These variations in the accuracy of the predictions depend mostly on the artificial neural networks architecture used, the regularity of the building operation and the accuracy of data measurement devices.

In 1993, in a study by Ansett and Kreider [2], building utility measurement data from a university campus centre, including electricity, natural gas, water and steam use, were modeled. The study considered weather, building occupancy and activity as the independent variables. Back-propagation architecture was used in this effort. The main focus was on testing different training methods, layering and data input order. The study presented an encouraging potential for the application of neural networks in building energy modeling. The study also stated the need for future investigation in selecting more accurate and effective learning algorithms.

In 1993, Curtiss et al. [3] used artificial neural networks to optimize energy consumption on an HVAC system. In this approach, the weather and building occupancy were considered as independent variables, and the HVAC system setpoints such as mixed air temperature, chilled water temperature, duct static pressure and chilled water flow rate were considered as dependent variables. Optimum setpoints were identified by varying the dependent variables that would yield the minimum electricity consumption. The building data was generated by using an HVAC Laboratory. The results of this study showed the need to apply the model to larger sized buildings with actual building measurement data, in order to validate the artificial neural networks method's efficiency.

In 1995, Cohen and Krarti [4] used energy consumption data generated from the DOE-2.1E Building Energy Analysis Program as input to the artificial neural networks model developed. The model was based on multi-layered feed-forward networks. This study mentioned the potential use of artificial neural networks methods in building energy savings estimates and recommended that future artificial neural networks modeling studies be done based on real building measurement data.

In 1995, Kreider et al. [7] investigated the prediction of future building energy consumption and system identification without the knowledge of immediate past energy consumption. Recurrent neural networks were used in the modeling. According to the authors, the recurrent networks offer an accurate method for predicting hourly energy use well into the future for thermal end uses when only weather data is known. During network training, actual measured data from a few past hours were used as input to the

model. However, during the prediction period, the network's own outputs were cycled back into the inputs. The building energy data for this model was also generated from the DOE-2.1E Building Energy Analysis Program. Although the error rate was relatively higher in this method when compared to, for example, the back-propagation method, it was still presented as an applicable method in predicting the future building energy use for retrofit energy savings estimation purposes. This study also stated the need for future study based on real building measurement data.

In 1996, Chonan et al. [9] applied Bayesian neural network for estimating building energy use. In this method, the known relationship between the input variables and output was used in combination with the neural network training. Jang et al. [10] used an auto-associative neural network in predicting missing building input–output data based on feed-forward network identity mapping. This method is effectively used when the building data has been available for some periods of time and missing for other periods of time. The noise filter capabilities of auto-associative neural networks proved to be effective in preprocessing the model data.

In 1996, Curtiss [8] described the use of neural networks in continuous control of feedback loops in an HVAC system and overall building energy use prediction. In this method, the input and output training data set were updated with new input data and a neural network output prediction from one previous time segment. The training data set was renewed with the latest building information and kept current for the near future predictions. Additionally, in this study, Curtiss used the neural network control algorithm

along with the traditional PI control algorithm to develop the optimum control parameters and enhance control capabilities of both methods.

In 2000, Breekweg et al. [6] evaluated a number of artificial neural networks techniques in the development of a generalized method for building energy-related fault detection. Real-time data from four different buildings and simulation data from one building were modeled based on a normalized radial basis function, specifically the general regression neural network as the normalized radial basis function was used. The coefficient of variation was higher, in the range of 20-40% for most buildings, except two buildings, which were in the range of 4-8%. The large deviations in the results were attributed to the quality of data measurement, building operation consistency and minimization of the noise elements in the data set. This study also reported the necessity to test the developed artificial neural networks model with energy data from different buildings in order to ensure the generalizing capacity of the model.

2.3 Case Study

2.3.1 Application of Neural Networks for the Prediction of Energy Consumption in a Supermarket

In 1997, a study by D. Marriott, D. Datta and S. A. Tassou [11] addresses the performance of a neural network in the prediction of electricity demand in a supermarket. It evaluates the capability of a neural network to forecast the overall power consumption

of the store every half hour with respect to time of day and environmental conditions. Three layered feed-forward neural networks were trained using the actual measured data collected from the store.

The input variables (input nodes) used in each network configuration are listed in Table 2-1.

Table 2-1: Input variables used for each network configuration for case I

Network 1	Day, Time, External Humidity and Temperature and Internal humidity and temperature for short term i.e. a month
Network 2	Day, Time, External and Internal Humidity for a month
Network 3	Day, Time, External and Internal Temperature for a month
Network 4	Day, Time, External Humidity and Temperature for a month
Network 5	Day, Time, Internal Humidity and Temperature for a month
Network 6	Day, Time, External Humidity and Temperature over long term i.e. four months
Network 7	Time-Series Prediction using past six time steps
Output Variable	Electrical Power Consumption in kW

The networks varied in terms of the number of input variables, i.e. input nodes, n. The number of nodes in the hidden layer varied as a function of the input nodes as (2n + 1). The standard back-propagation algorithm was employed to train all the networks. Back-propagation is an integrative training algorithm designed to minimize the mean square error between the output of the network and the actual value. A sigmoid function is used

for the transfer function as it enables a finite number of nodes in the single hidden layer to uniformly approximate any continuous function.

The results indicated that using a short-term data-set, i.e. the previous month's data, may be adequate to accurately predict half hourly electrical demand in retail food stores. A combination of time series and multiple independent variables modeling may improve the performance of the network in regions of seemingly random fluctuations. The time of day is the most significant independent variable with a high percentage contribution. A comparison of the prediction performance of the network against more traditional statistical approaches is also presented. Artificial neural network modeling provides a much better prediction of electrical energy consumption than regression modeling. The correlation coefficient for artificial neural network modeling varies from 0.91693 to 0.95499, and correlation coefficient for regression modeling varies from 0.48673 to 0.74360. Further work needs to involve the development of artificial neural networks to predict electrical energy on-line and also predict the energy consumed by the various subsystems.

2.3.2 Neural Networks for Energy Flow Prediction in a Hospital

In 1999, L. Frosini and G. Petrecca [12] studied the short-term prediction of the thermal energy consumption of a hospital. Non-linear models based on Multi-Layer Perceptron feed-forward neural networks have been implemented as a second step of the procedure, only after the linear estimate of the best regression vectors. This procedure allows

evaluating a greater number of models compared to a procedure starting with the neural models estimate.

The input variables (input nodes) used in each network configuration are listed in Table 2-2.

Table 2-2: The best ten ARMA models

Network	Inputs	n_a	n_b	SSR
1	Tex	4	4	39.24
2	Tex	5	5	37.94
3	ΔT1	2	2	38.1
4	ΔT1	3	3	38.87
5	A, T1	2	2	39.64
6	A, ΔT1	2	2	37.50
7	A, ΔT4	2	2	39.95
8	Tex, T1, T4	2	2	39.50
9	Tex, T1, T4	4	4	39.90
10	A, Tex, T1, T4	2	2	39.34

where:

Tex = external temperature,

TI, T4 = internal temperature (zone 1, 4),

ΔTI, ΔT4 = (Tex - TI), (Tex -T4).

n_a is equal to the number of poles of the system transfer function,

n_b is the number of zeros of the system transfer function. (See the corresponding predictor and the regression vector below)

SSR: Sum of Squared Residuals [31] is a measure of the discrepancy between the data and an estimation model:

$$SSR = \sum_{i=1}^{n} (y_i - f(x_i))^2$$

where:

y_i is the regression model,

$f(x_i)$ is the data.

A: An ARMA (Auto Regressive Moving Average) model is described by the following equation:

$$A(q^{-1})y(t) = B(q^{-1})u(t) + e(t)$$

where:

y is the output of the dynamic model,

u is the input,

e is the disturbance or noise,

q^{-1} is the shift operator.

The corresponding predictor:

$$\hat{y}(t) = -a_1 y(t-1) - \ \ldots - a_{n_b} y(t-n_a) + b_1 u(t-1) + \ \ldots + b_{n_b} u(t-n_b+1)$$

is thus based on the regression vector:

$$\Phi(t) = [y(t-1), \ldots, y(t-n_a), u(t), u(t-1), \ldots, u(t-n_b)]^T$$

where:

n_a is equal to the number of poles of the system transfer function,

n_b is the number of zeros of the system transfer function.

The considered variables are: natural gas consumption, cold and hot water consumption, internal temperatures (recorded in four different zones) and external temperature.
In addition to the recorded variables, they have considered four other variables; the differences between the internal temperatures and the external temperature (ΔT).
They have selected the best linear models through the cross validation, i.e. they have identified the models by the Least Mean Squares estimate using the training data set and hence they have chosen the models that provided the lowest values of the Sum of Squared Residuals (SSR) during the validation stage.

On the basis of these ten ARMA models, they built the same number of Neural Network Auto Regressive Moving Average NNARMA models using a Multi-Layer Perceptron network. The Multi-Layer Perceptron network has three layers: an input layer, an output layer and a hidden layer. They used hyperbolic tangent functions for the neurons in the hidden layer and linear functions for the neurons in the output layer.

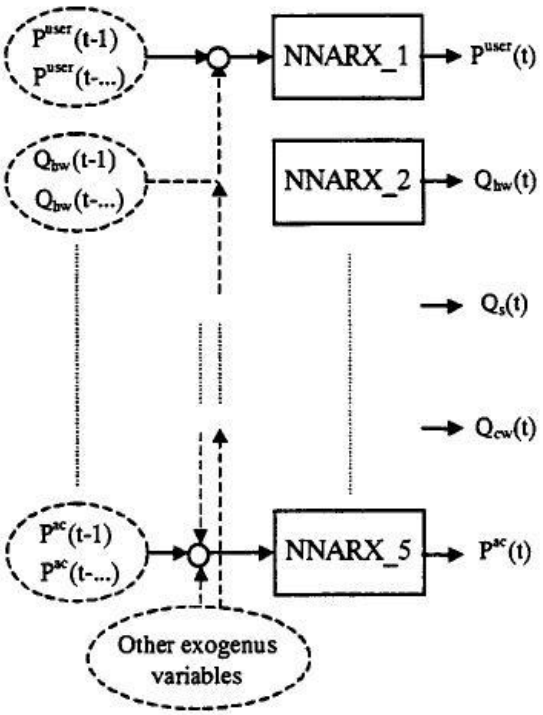

Figure 2-2: Prediction of energy flows

The predicted output $\hat{y}(t)$ can depend on the past values y (t-k), where

k= 1, 2, ...n_a, and possibly on the past values of input u(t-k), where k = 1, ... n_b.

$$\hat{y}(t) = g[(y(t-1), y(t-2) \ \dots y(t-n_a), u(t), u(t-1) \dots u(t-n_b)]$$

The output is the energy flows, such as electric energy consumption P_{user} , hot water consumption Q_{hw}, steam consumption Q_s, chilled water consumption Q_{cw} and air compressed consumption P^{ac} .

The results obtained from the Auto Regressive Moving Average(ARMA) and feed-forward neural networks (NNARMA) models are compared, concluding that NNARMA

models provide better results than ARMA models, but the analysis of ARMA models is necessary to obtain guidelines in the choice of the best regression vector as input for neural models. Further improvements could be brought by a different choice of the training and validation sets or by the use of recurrent neural networks.

2.3.3 Artificial Neural Networks Applications in Building Energy Predictions for Tropical Climates

In 2005, Melek Yalcintas and Sedat Akkurt [13] studied the building energy predictions for tropical climates, the power consumption of the central chiller plant, including the chillers, cooling tower and pumps, was modeled based on the three layer feed-forward artificial neural networks method. The climate data variables included dry bulb temperature, wet bulb temperature, dew point temperature, relative humidity percentage, wind speed and wind direction. The input variables and output variable are listed in Table 2-3.

Table 2-3: The data used in ANN model construction for case 3

Inputs	Output
Time (hour)	Total building power consumption
Dry bulb temperature	
Wet bulb temperature	
Dew point temperature	
Relative humidity	
Wind speed	
Wind direction	

In this study, the common three layer feed-forward type of artificial neural network was used. The input layer, hidden layer and the output layer contained 7, 6 and 1 neurons, respectively. Neurons in each layer are completely connected to each neuron in the

neighboring layer. No bias or momentum term was used in the generation of the model. The original data was composed of 121 sets each with 7 input parameters, and 1 output parameter. The input parameters and output parameter are given in Table 2-3. The data was first split into two parts to use for training (80 sets) and testing (41 sets) of the model. 2000 iterations were performed in this study to train the model. The transfer function used in this study was a sigmoid function with a standard back-propagation algorithm.

The model was successfully created and was able to closely agree with the actual measured data. The artificial neural networks model developed in this study processes data from two main origins: climate and HVAC system in Hawaii. Therefore, with the help of this model, they are able to predict the chiller plant power consumption as a function of meteorological parameters like the wind speed, wind direction, dry bulb and wet bulb temperatures and relative humidity. Air conditioning processes are controlled within narrow parameter ranges in Hawaii because the annual climate variation is narrow. Hence, the data range employed in this model construction was narrow. This has resulted in a model that can predict within a narrow range. The unique climate consistency in Hawaii indicates that there is a greater chance that the artificial neural networks methods can be successfully implemented in building energy predictions.

2.3.4 Summary

The artificial neural networks models in the work cited here have used building energy data from building simulation, laboratory experiments and actual building measurement data. While for the sake of simplicity the simulation data in the initial artificial neural

networks modeling stages are useful, it is essential to use actual building data during the later development stages to account for the possible imperfections in the measured data. Also, the actual building data is the best indicator of the building features, operation and equipment efficiency.

Chapter 3: Proposed Artificial Neural Networks Models

3.1 Introduction

Artificial neural networks are good for cases where you do not easily know how the input and output relate –but, you do know whether the output is right or wrong [14]. The artificial neural network approach is a generic technique for mapping the relationship between inputs and outputs; it requires less expertise and experimentation than traditional modeling of non-linear multivariate systems. The neural network learns the input/output mapping of a system through an iterative training process. It can also update its learned knowledge on-line over time. This automatic learning property makes a neural network based system inherently adaptive.

In this thesis, artificial neural network models used actual building data. Repeated building data measurements from different buildings were used also in developing the artificial neural networks models. The past values were added as one of the inputs to improve the prediction of proposed artificial neural networks models. The three layered feed-forward neural networks were trained using the actual measured data collected from the building. The network size varied at the beginning. The standard back-propagation algorithm was employed to train all the networks. Back-propagation is an integrative training algorithm designed to minimize the mean square error between the output of the network and the actual value.

The artificial neural networks models in this project contained several different buildings:

Table 3-1: The buildings used in ANN models

Building number	Name of building
1	Administration
6	Performing Art Center
10	Alan A. Erhart Agriculture
21	Engineering West A
21	Engineering West B
35	Robert E. Kennedy Library
40	Engineering South
43	Recreation Center
113	Sierra Madre Hall
114	Yosemite Hall
192	Engineering IV
N/A	Mustang Substation Main
N/A	The entire campus

With the proposed research, the engineers of the Engineering and Utilities Facility Services of Cal Poly can predict future energy consumption. Using the predicted power consumption from the proposed models, they can determine the most effective project to pursue. They can also predict the power consumption of a new building based on a similar building.

3.2 Artificial neural networks model input and output data

The current research will predict the power consumption for 12 different buildings and the entire campus. Neural networks were used in the prediction of power consumption of building 1 at Cal Poly, the Administration building. November 2008 was chosen as it contained the least amount of error.

The power data contains 30 days with 2877 points of values of power consumptions in kW. The time intervals between each point are 15 minutes. Because the period is so long, the time is shifted. The weather data contains 5098 points, including date, time, temperature in F, dew point in F, pressure in inch, wind speed in mph, wind gust in mph, humidity in % and rainfall rate (hourly). The time intervals between each point are not regular. Some of them are 5 minutes, others are 10 minutes or 15 minutes, and the time is shifted as well.

The power data and weather data needed to be sampled in order to merge them together. The nearest-neighbor interpolation was used to sample the data into every 15 minutes. Nearest-neighbor interpolation [15] is a simple method of multivariate interpolation in 1 or more dimensions. Interpolation is the problem of approximating the value for a non-given point in some space, when given some values of points around that point. The nearest-neighbor algorithm simply selects the value of the nearest point, and does not consider the values of other neighboring points at all, yielding a piecewise-constant interpolant. The downside of the nearest-neighbor interpolation is it will generate ‘not a number’ for points that are out of the range. These points were removed in order to prevent them from getting into the neural networks.

The original data was composed of 2,878 sets, each with 8 input parameters and 1 output parameter. The inputs and output were given in Table 3-1. The data was first split into two parts to use for training (2,000 sets) and testing (878 sets) of the model.

Table 3-2: The data used in ANN model

Input	Output
Temperature in F	Building power consumption in kW
Dew point in F	
Pressure in inch	
Wind speed in mph	
Wind gust in mph	
Humidity in %	
Rainfall Rate (Hourly).	
Previous building power consumption in kW	

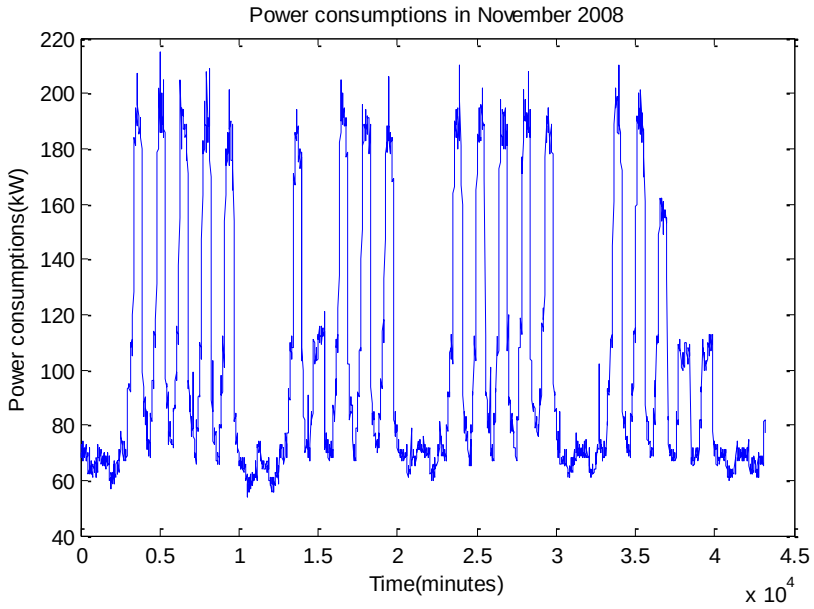

Figure 3-1: Power consumptions in November 2008 of Administration Building

3.3 Artificial Neural Networks Model Architecture

After the data was processed, the most suitable artificial neural networks architecture would be determined for the proposed models. This led to the question of how many hidden layers and hidden neurons to use?

If there is only one input, there seems to be no advantage to using more than one hidden layer. However, things get much more complicated when there are two or more inputs. The complexity of a neural network will depend on the following factors [16]:

1. The numbers of input and output units.
2. The number of training cases.
3. The amount of noise in the targets.
4. The complexity of the function or classification to be learned.
5. The architecture.
6. The types of the transfer functions in hidden layer.
7. The training algorithm.
8. Regularization.

In most situations, there is no way to determine the best number of hidden units without training several networks and estimating the generalization error of each. If there are too few hidden units, a high training error and high generalization error will be generated due to under-fitting and high statistical bias. If there are too many hidden units, a low training error will be generated but still have high generalization error due to over-fitting and high

variance. Geman, Bienenstock, and Doursat [17] discuss how the number of hidden units affect the bias/variance trade-off.

The simulation started with the common three layer feed-forward type of artificial neural networks. A main principle in identification is to “try simple things first.” The idea is to start with the simplest model which has a possibility to describe the system and continue to more complex ones if the simple model does not provide reliable results in the validation stage [18]. When a more complex model is investigated, the results with the simpler model give some guidelines how the structural parameters should be chosen in the new model. The artificial neural networks architectures were shown in Table 3-3.

Table 3-3: ANN model architecture

Network	Type	Number of nodes in input layers	Number of hidden layers	Number of nodes in the first hidden layer	Number of nodes in the second hidden layer	Number of nodes in output layer
1	feed-forward	8	1	9	NA	1
2	feed-forward	8	2	10	10	1
3	feed-forward	8	2	10	7	1
4	feed-forward	8	2	10	5	1

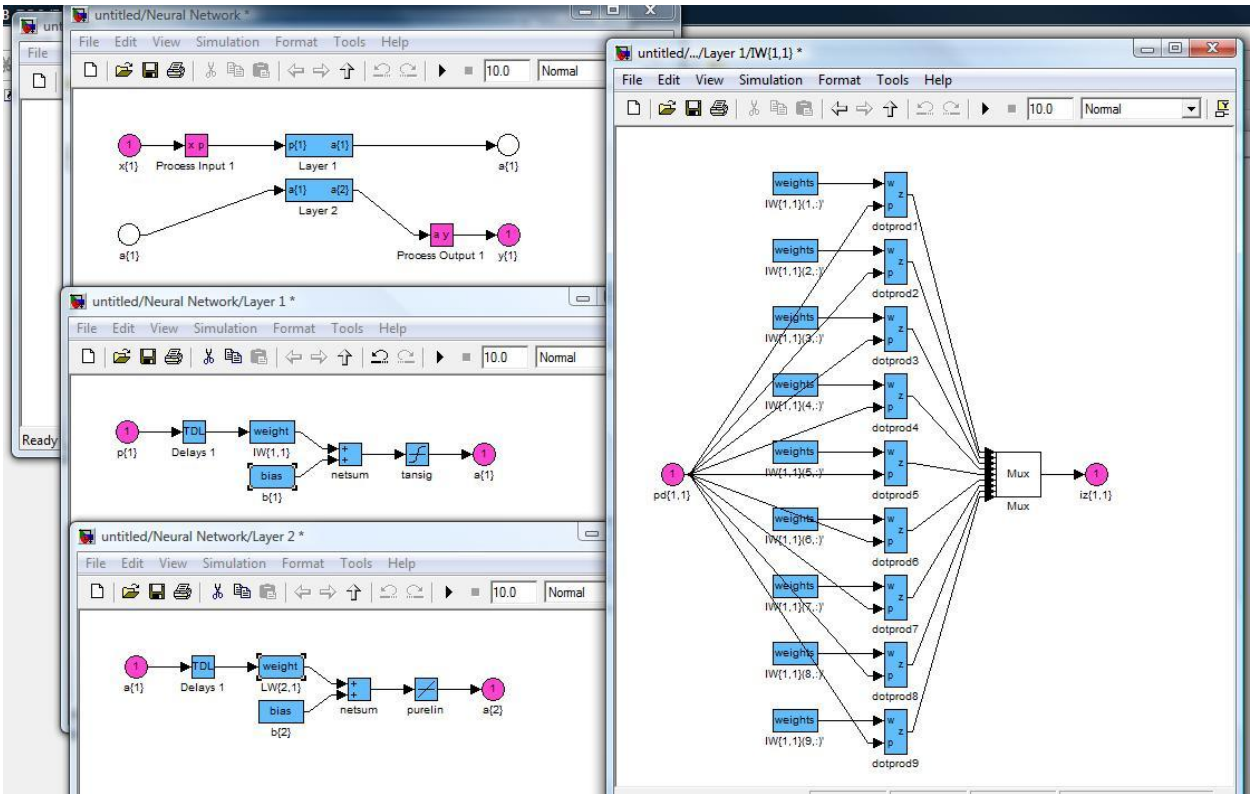

Figure 3-2: The ANN model architecture used for network 1

The block diagrams shown in Figure 3-2 are generated from simulink after the simulation. The diagram on the top left hand corner shows the overall artificial neural networks architecture for network 1. The artificial neural network is a non-linear mapping of the space between an input data set and an output data set and consists of three parts - an input matrix P (independent variables), an output vector Y (dependent variables), and an algorithm that maps the input space to the output space. One or more hidden layers connect the external layers by a set of "weights", expressed as two-dimensional matrices, W. In a feed-forward neural network, the value of each node in a particular hidden layer is the result of the transfer function whose argument is the weighted sum over all the nodes in the previous layer plus a constant bias B.

The diagram in the middle of the left is the hidden layer. The diagram at the bottom left hand corner is the output layer. The diagram on the right shows the data flowing and summations of weighted inputs.

The output of the hidden layer:

$$y_h = \text{f}\,[\text{W} * \text{P} + \text{B}]$$

Transfer function

The transfer function is an output function that converts the net input value to the node output value [14]. The sum of the weighted inputs and the bias forms the input to the transfer function. The hyperbolic tangent sigmoid transfer function was chosen for hidden layers; and the linear transfer function was chosen for output layer. They were chosen because a linear output neuron can allow the output to take on any value. They are the most commonly used transfer functions for back-propagation.

Table 3-4: Transfer function

Function	**Definition**	**Range**
Linear	x	$(-\infty,+\infty)$
Hyperbolic tangent	$\frac{1-e^{-x}}{1+e^{-x}}$	(-1,+1)

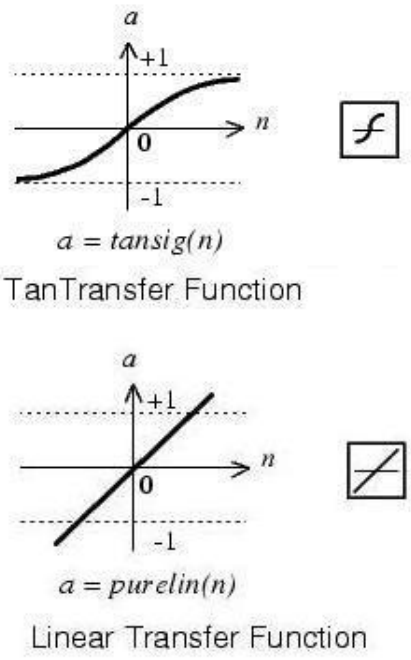

Figure 3-3: Transfer function [19]

3.4 Training and Testing of Artificial Neural Networks

Training is the process of adjusting the weights and bias [14]. Three and four layered feed-forward neural networks were trained using the actual measured data collected from the buildings. The networks varied in terms of the number of hidden layers and number of hidden neurons, as shown in table 3-2. The standard back-propagation algorithm was employed to train all the networks. Back-propagation is an integrative training algorithm designed to minimize the mean square error between the output of the network and the actual value.

Training protocol

Assign weights with random values

Loop until the last iteration is reach {

1. send the network some input.
2. compare the output of the network with the correct output
3. if the output is wrong, use back-propagation to adjust the weights and the bias.

}

Once the network is trained, the weights are set and you just use the network (no more weight adjustment)

Main idea of back propagation

- first assign random numbers to the weights and bias (between 0 and 1)
- compute the error at the output layer neurons
- compute the error at the hidden layer neurons
- iterate until results are acceptable

1. For each output neuron K, compute the error
   - the actual output of neuron K at some iteration p is $y_k(n)$
   - the desired output is $y_{dk}(n)$
   - the error at iteration n is $e_k(n) = y_{dk}(n) - y_k(n)$
2. Adjust the weights in the output neurons
3. Adjust the weight of the hidden layer neurons

Learning

In this thesis, the standard back-propagation is employed to train the neural network. It calculates the weight change Δw for a given neuron based on gradient descent:

$$w_{ij}(n+1) = w_{ij}(n) + \Delta w_{ij}(n)$$

$$\Delta w_{ij}(n) = \propto * \ y_i(n) * \ \delta_j(n)$$

where:

$\propto$ is the learning rate,

$y_i(n)$ is the output of neuron j,

$\delta_j(n)$ is the error gradient.

The learning rate is chosen as 0.01.

Chapter 4: Results

4.1 Artificial Neural Networks Model Training Results

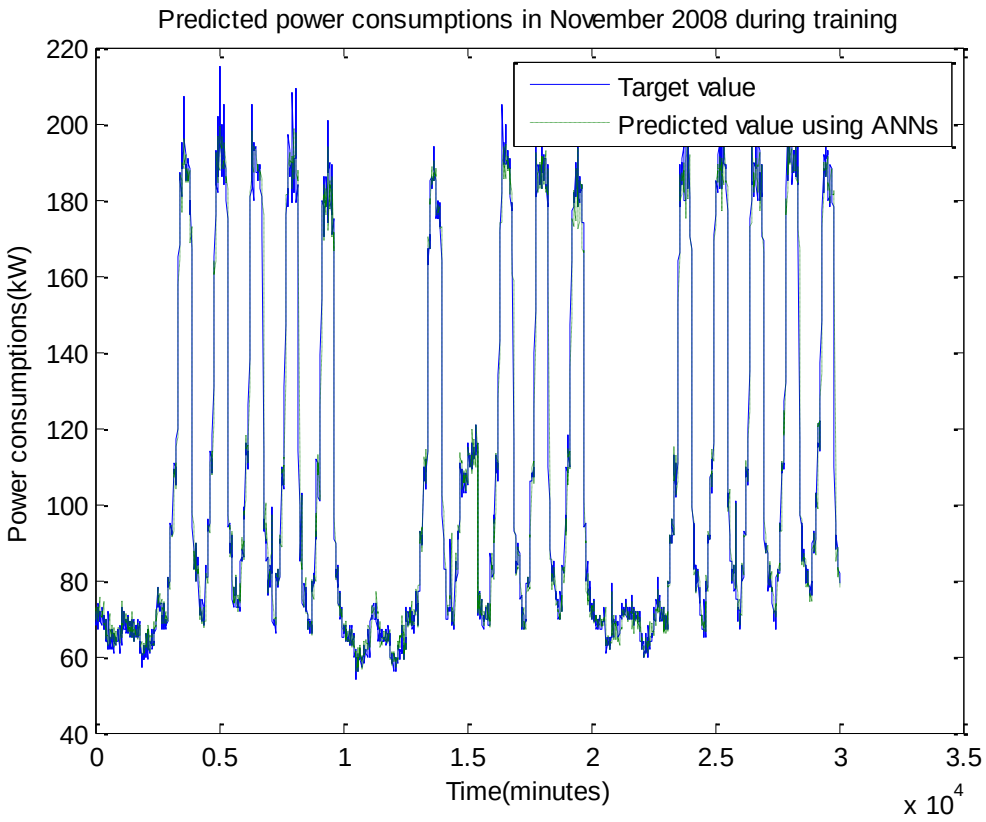

Figure 4-1: ANN model training results for network 1

The solid line is the actual power consumption in November 2008 during training. The dotted line is the predicted power consumptions in November 2008 during training.

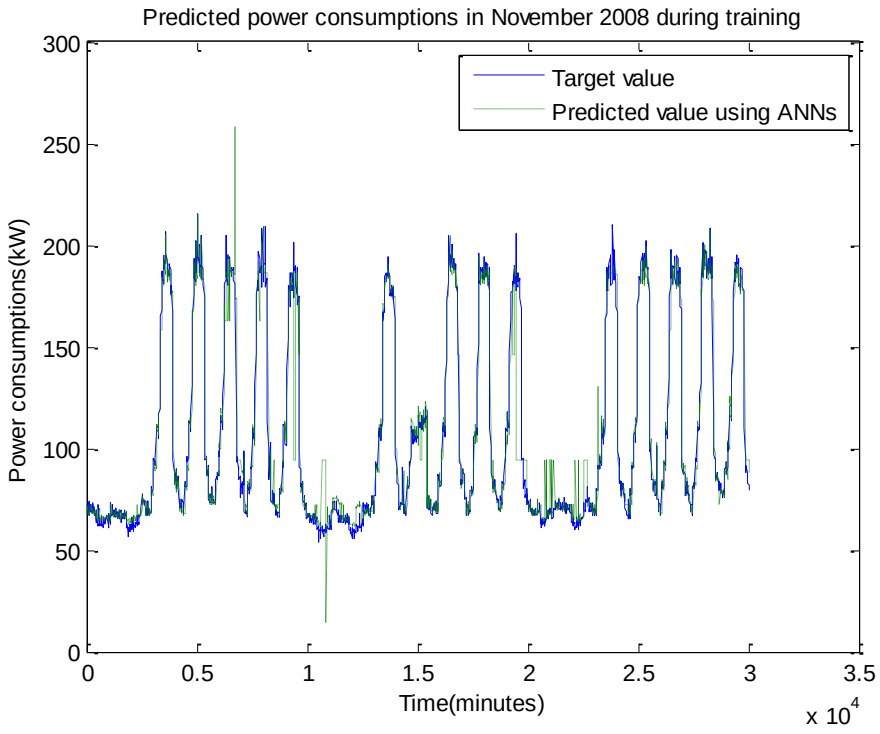

Figure 4-2: ANN model training results for network 2

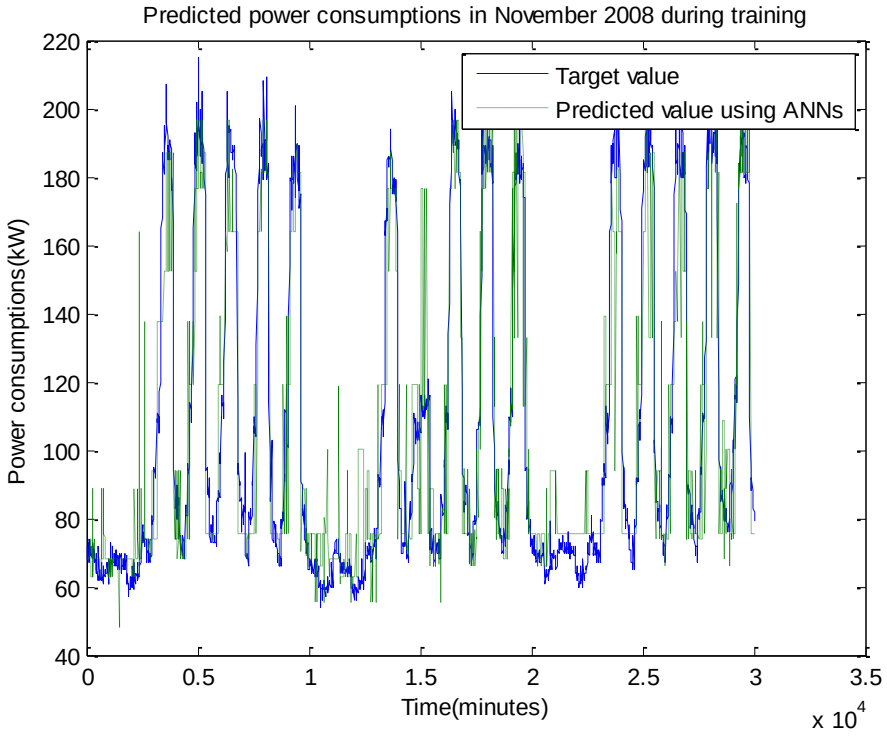

Figure 4-3: ANN model training results for network 3

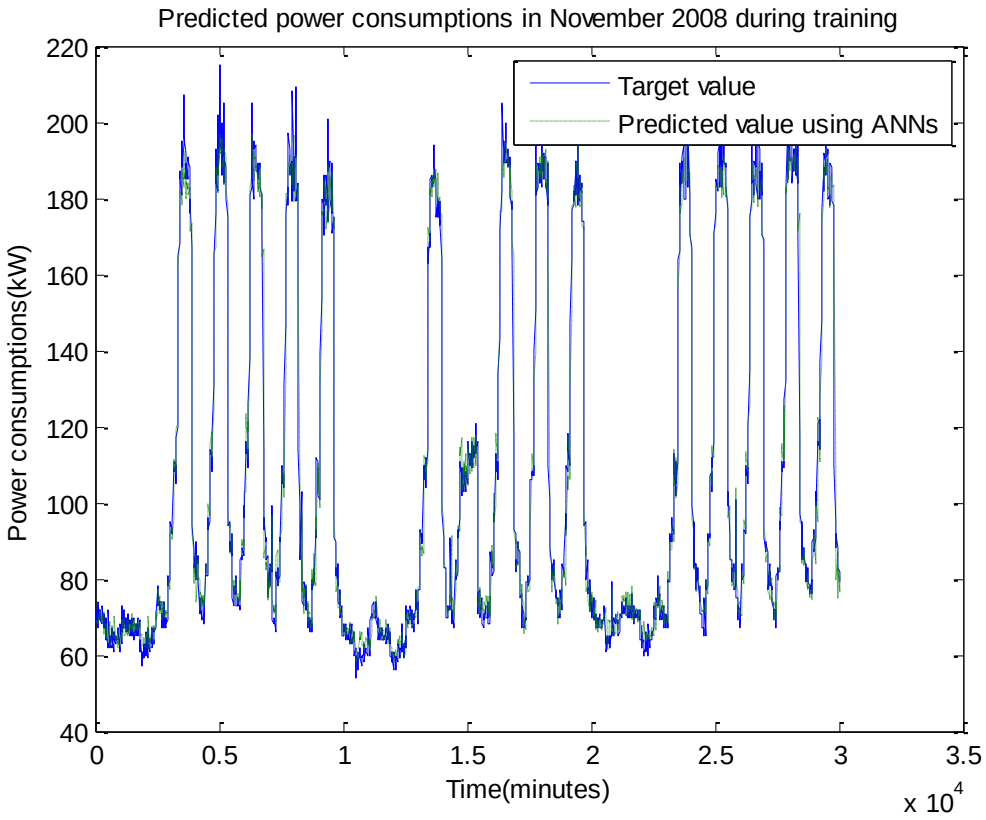

Figure 4-4: ANN model training results for network 4

As shown in Figure 4-1, the predicted power consumption of network 1 matches the actual power consumption. Figure 4-2 illustrates that the predicted power consumption of network 2 matches the actual power consumption as well. Network 2 seems as though it can trace the power consumption even closer, but it generates large target excursion. This may be caused by over-fitting. Figure 4-3 illustrates that the predicted power consumption of network 3 can only follow the general target value characteristics of the actual power consumption. This may be caused by under-fitting. Figure 4-4 illustrates that the predicted power consumption of network 4 matches the actual power consumption. After 50,000 iterations of training, the predicted power consumption of

network 1, 2 and 4 match the actual power consumptions. The performances of network 1 and 4 are very similar, but network1 has better performance on low values than network4. Overall, network 1 produces the best results.

4.2 Artificial Neural Networks Model Testing Results

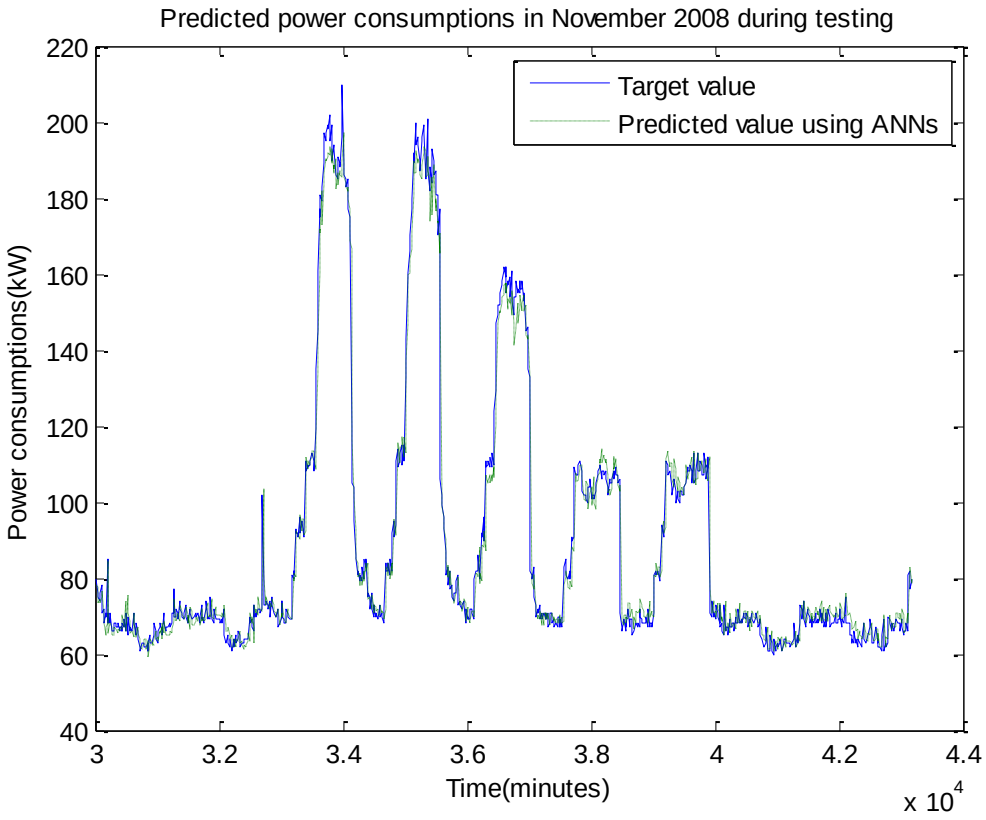

Figure 4-5: ANN model testing results for network 1

The solid line is the actual power consumption in November 2008 during testing. The dotted line is the predicted power consumptions in November 2008 during testing.

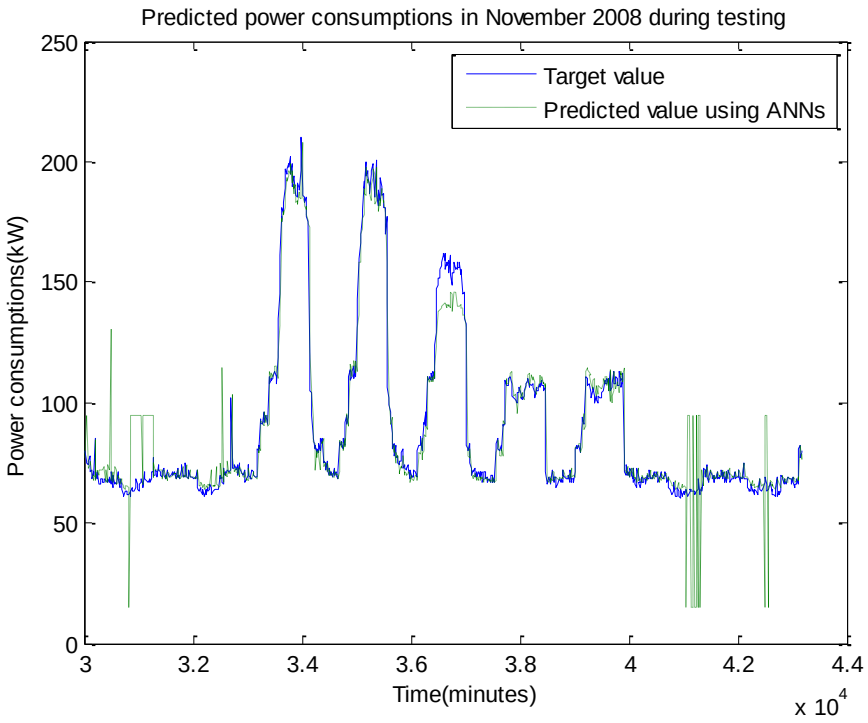

Figure 4-6: ANN model testing results for network 2

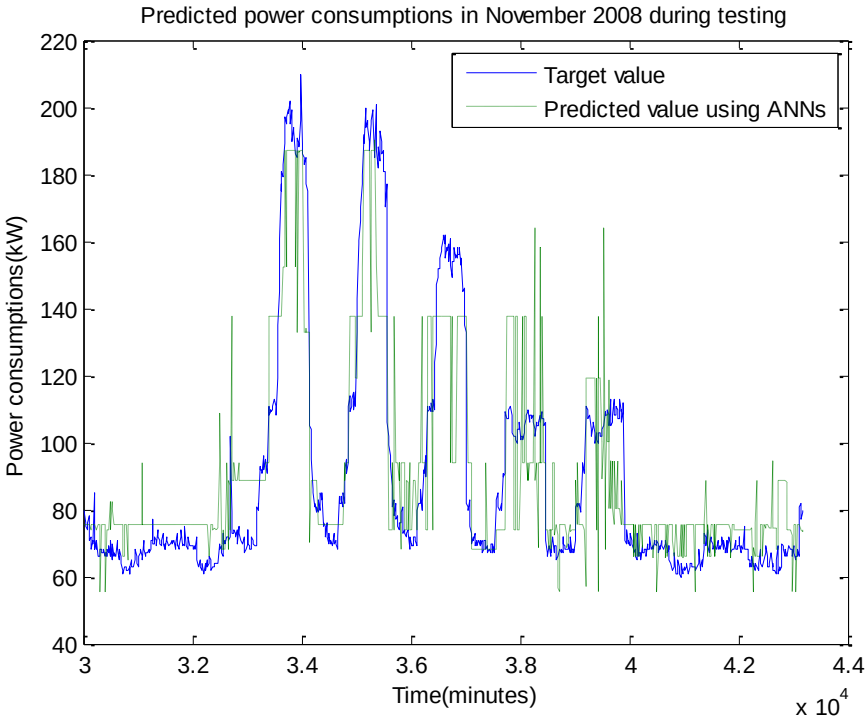

Figure 4-7: ANN model testing results for network 3

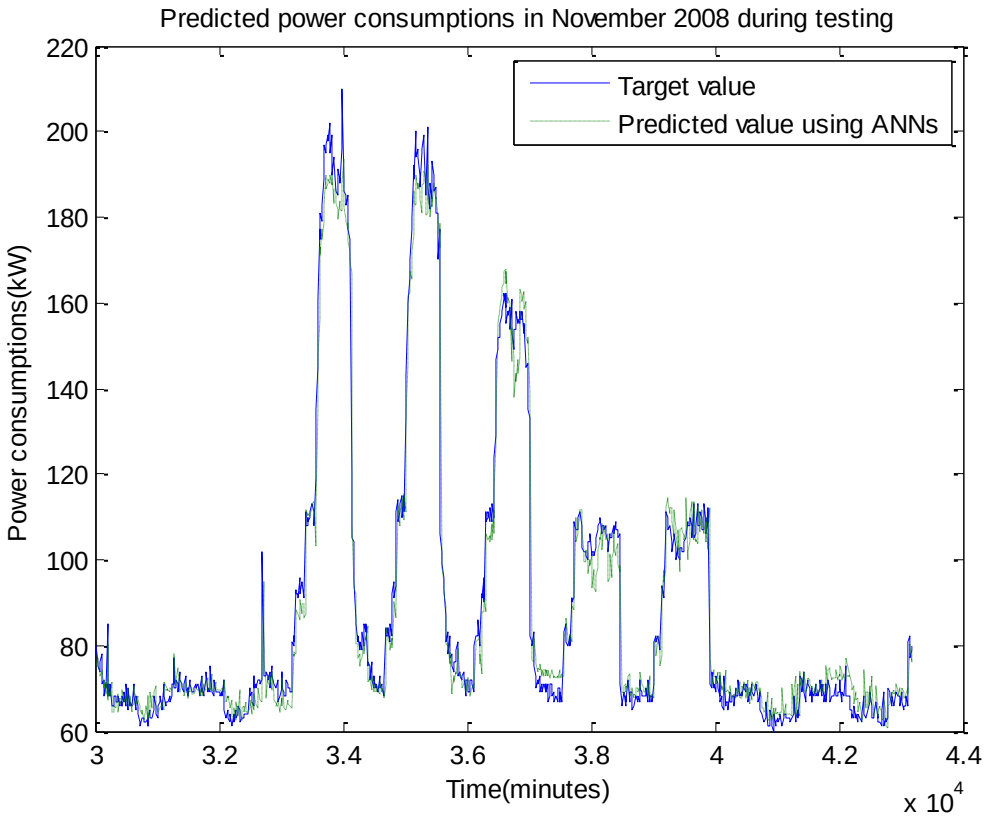

Figure 4-8: ANN model testing results for network 4

As shown in Figure 4-5, the predicted power consumption of network 1 matches the actual power consumption. Figure 4-6 illustrates that the predicted power consumption of network 2 matches the actual power consumption as well. Network 2 seems as though it can trace the power consumption even closer, but it generates large target excursion. This may be caused by over-fitting. Figure 4-7 illustrates that the predicted power consumption of network 3 can only follow the general target value characteristics of the actual power consumption. This may be caused by under-fitting. Figure 4-8 illustrates that the predicted power consumption of network 4 matches the actual power consumption. Same as the results of training, the predicted power consumption of

network 1, 2 and 4 match the actual power consumptions. The performances of network 1 and 4 are very similar, but network1 has better performance on low values than network4. Overall, network 1 produces the best results. Let's take a look at the details of the high peak and low peak.

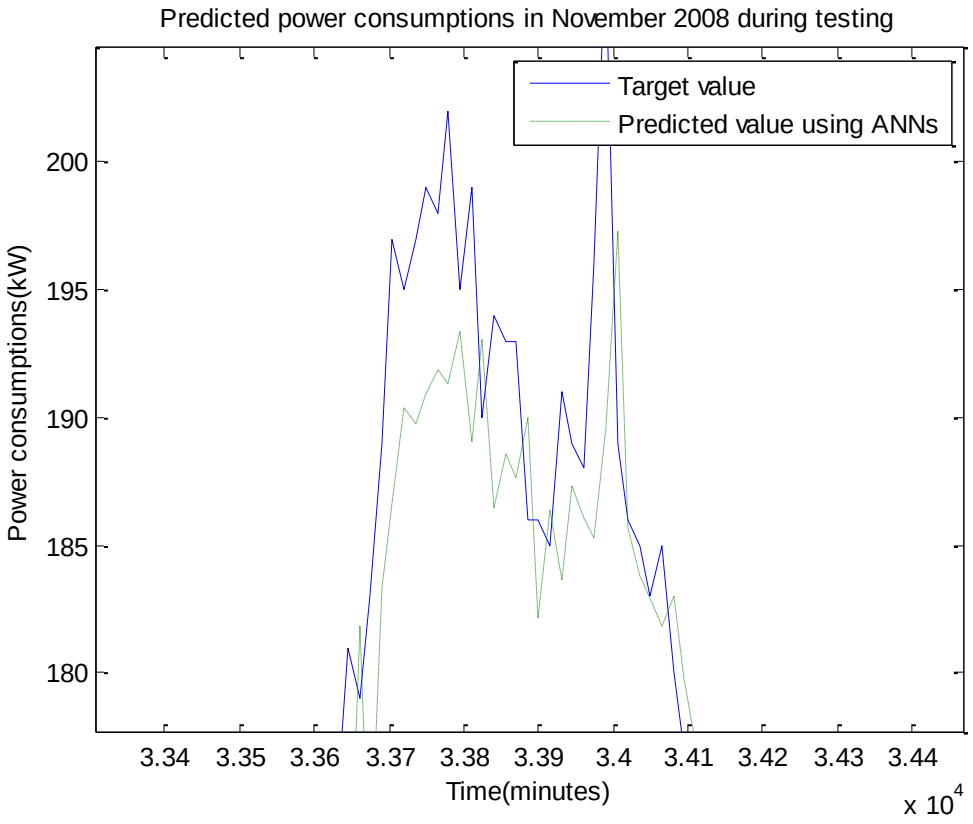

Figure 4-9: ANN model testing results at high peak for network 1

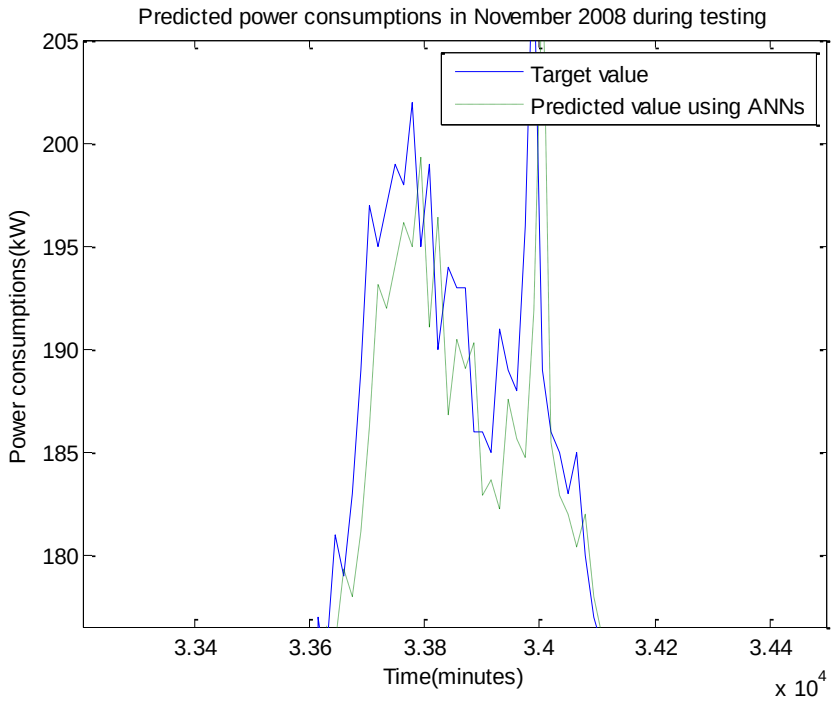

Figure 4-10: ANN model testing results at high peak for network 2

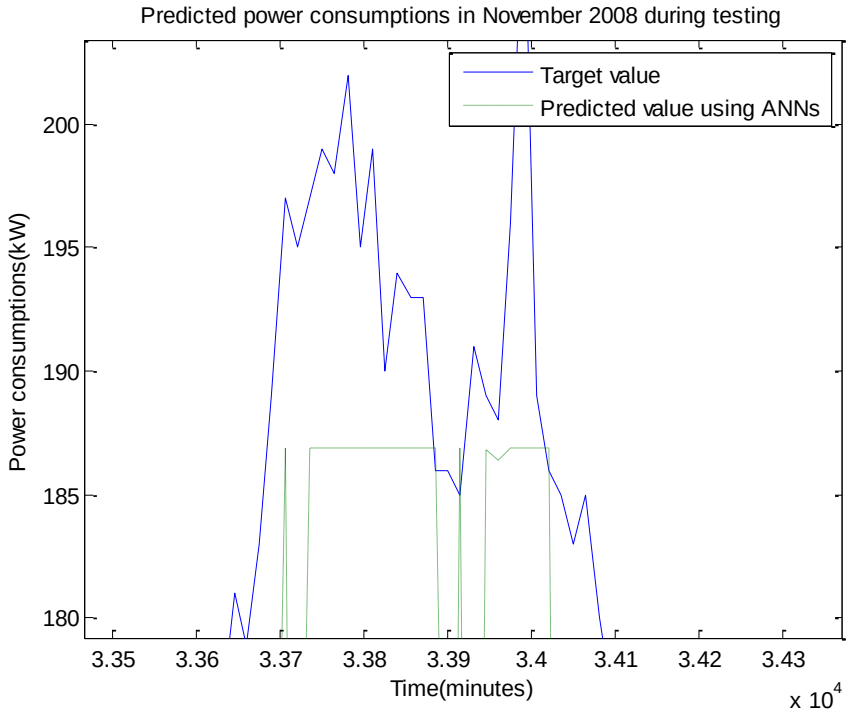

Figure 4-11: ANN model testing results at high peak for network3

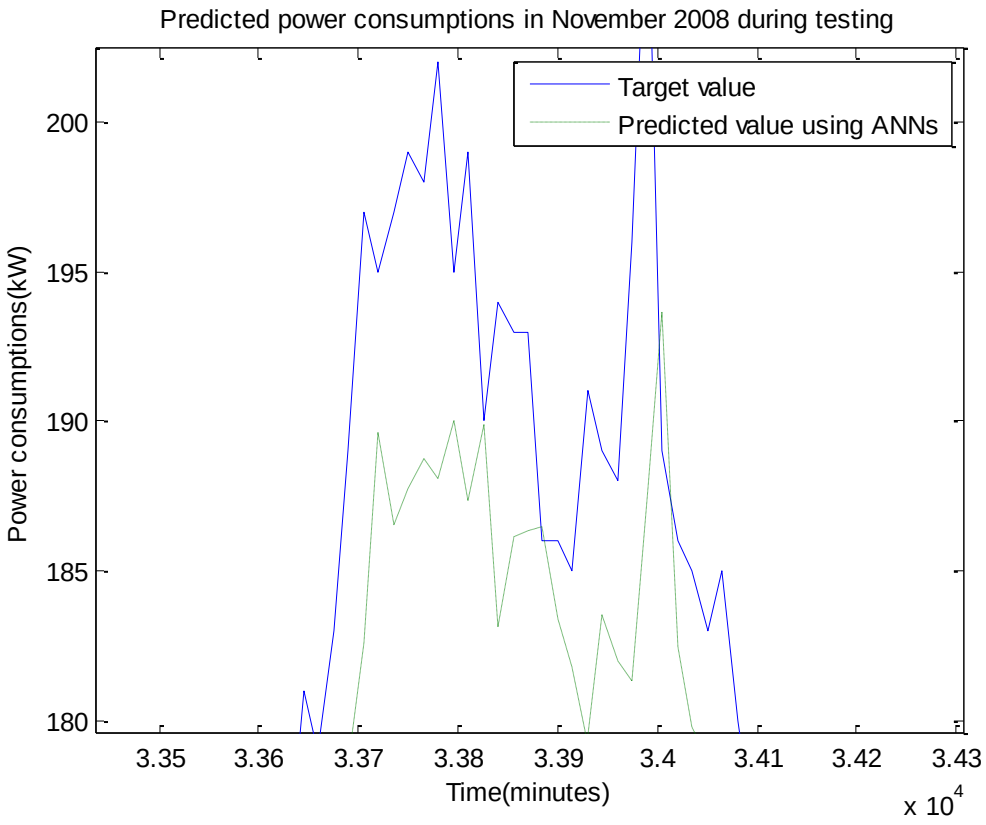

Figure 4-12: ANN model testing results at high peak for network4

As shown in Figure 4-7, the predicted power consumption of network 1 matches the actual power consumption at the high peak values. Figure 4-8 illustrates that the predicted power consumption of network 2 matches the actual power consumption at the high peak values as well. Network 2 seems as though it can trace the power consumption even closer. Figure 4-8 illustrates that the predicted power consumption of network 3 doesn't appear to follow the general target value characteristics of the actual power consumption at the high peak values; it just goes high and fails track the peak. It follows the general peak and valley trend of the overall consumption at the high peak values. This may be caused by under-fitting. Figure 4-9 illustrates that the predicted power consumption of

network 4 matches the actual power consumption at the high peak values. Network 2 has the best performance among the other networks at the high peak values. Network 1 produces the second best results and Network 4 produces the third best results. Network 3 is the worst.

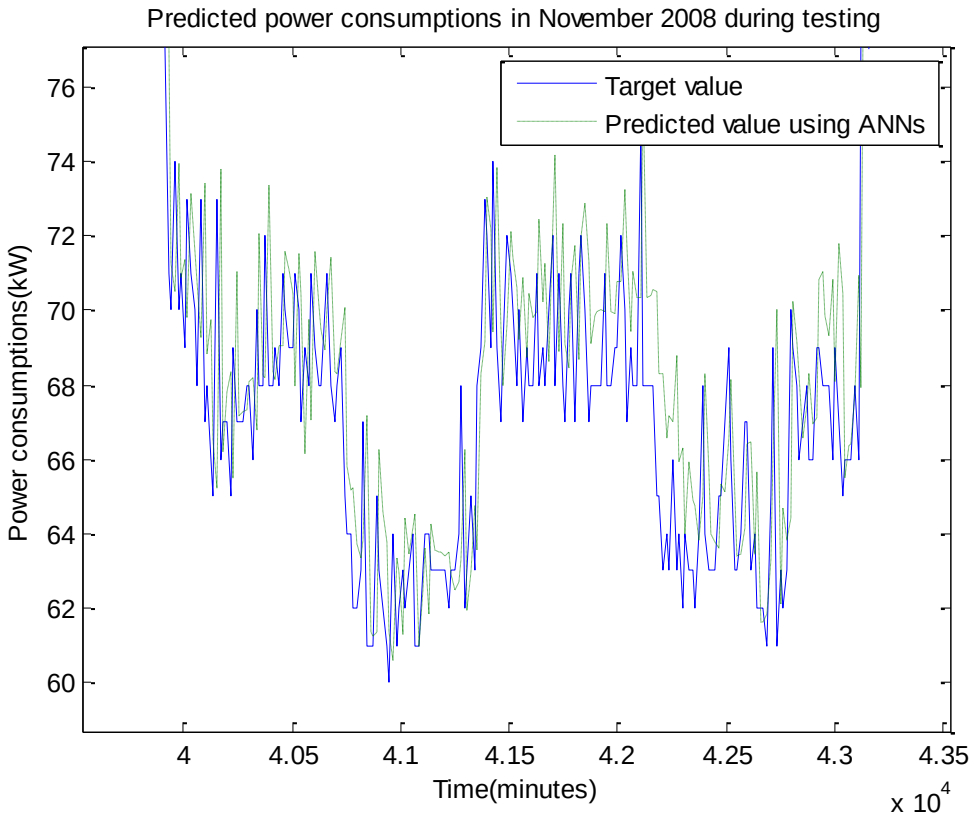

Figure 4-13: ANN model testing results at low peak for network 1

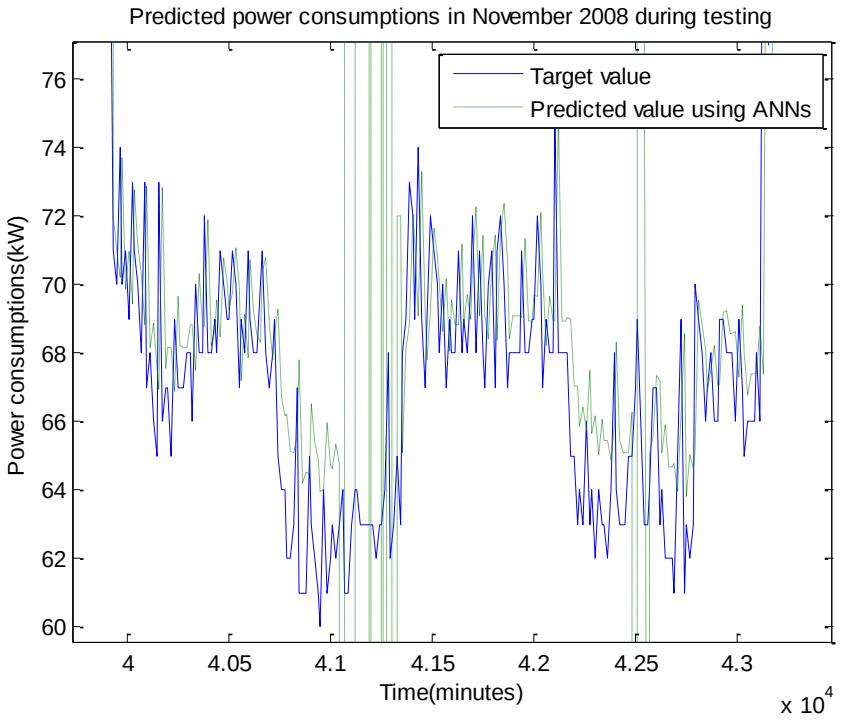

Figure 4-14: ANN model testing results at low peak for network 2

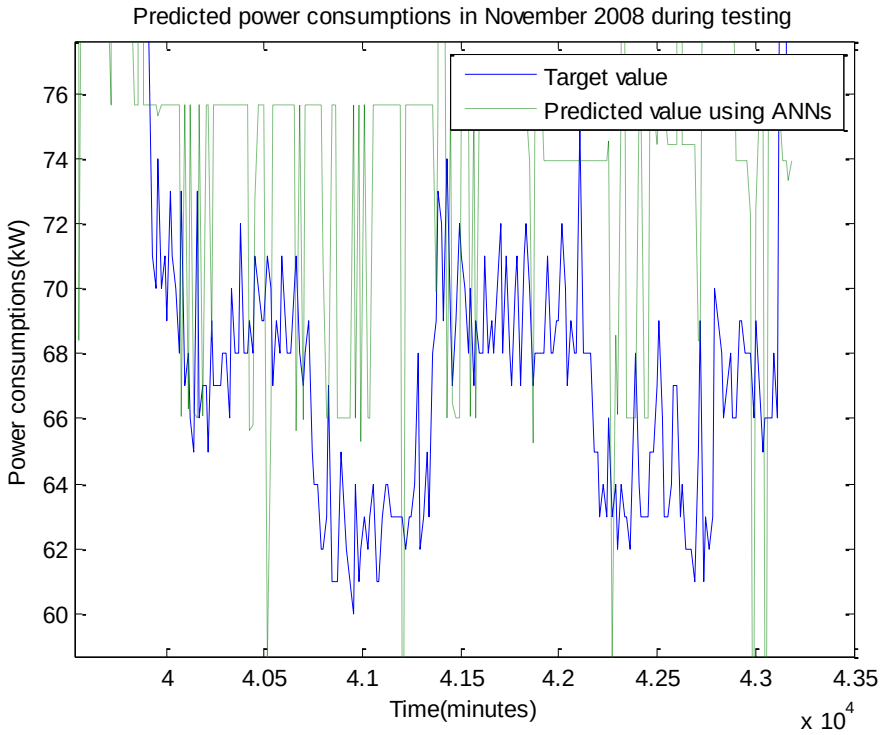

Figure 4-15: ANN model testing results at low peak for network 3

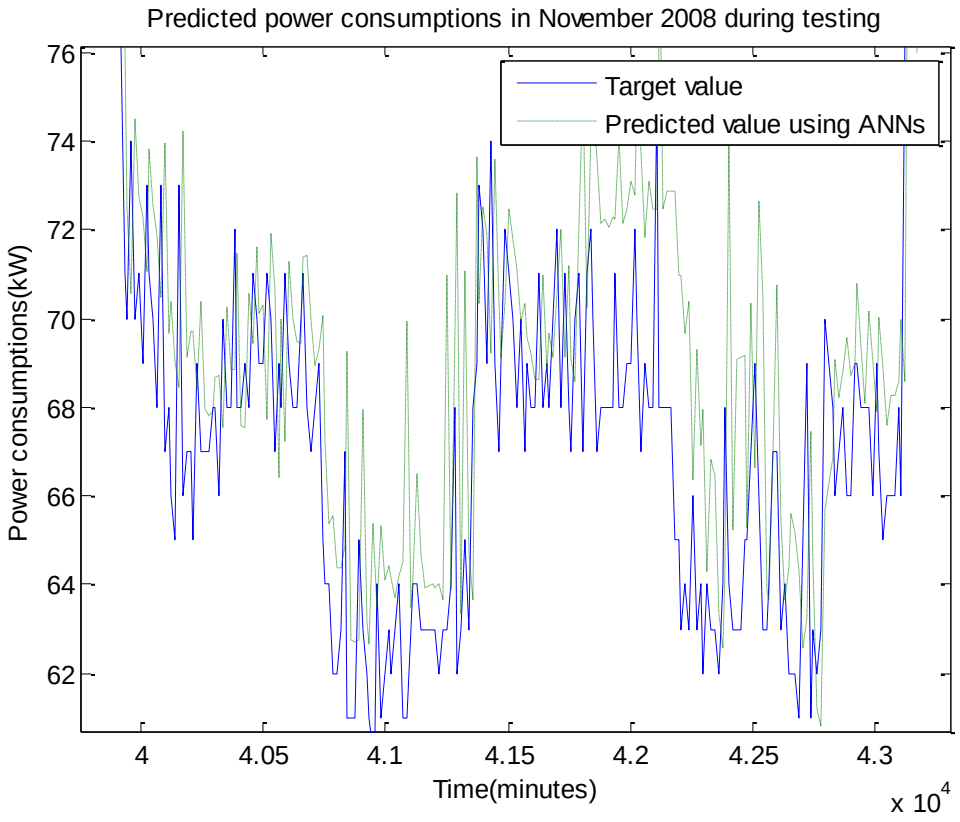

Figure 4-16: ANN model testing results at low peak for network 4

As shown in Figure 4-13, the predicted power consumption of network 1 matches the actual power consumption at the low peak values. Figure 4-14 illustrates that the predicted power consumption of network 2 matches the actual power consumption at the low peak values as well, but it generates large target excursion. Figure 4-15 illustrates that the predicted power consumption of network 3 doesn't appear to follow the general target value characteristics of the actual power consumption at the low peak values; it just drops low and fails track the valley. It follows the general peak and valley trend of the overall consumption at the low peak values. This may be caused by under-fitting. Figure 4-16 illustrates that the predicted power consumption of network 4 matches the actual

power consumption at the low peak values. Network 1 has the best performance among the other models at the low peak values. Network 2 produces the second best results and Network 4 produces the third best results. Network 3 is the worst. Among the performance of overall power consumptions prediction, high peak and low peak; network 1 rates the highest.

4.3 Comparison of Errors between each Network

Table 4-1: Statistics of power consumptions

Std Dev	Average	Maximum	Median	Mode
45.1368	101.5520	215.0000	80.0000	68.0000

Table 4-1 shows the statistic parameters of power consumption in November 2008. The power consumption of building 1 was varying from a range of 54.0000 kW to 215.0000 kW, and the average of power consumption was 101.5520 kW. We can use these parameters as a reference point to evaluate the performance of each network.

Table 4-2: Summary of network performance error

Network	Process	Std Dev	Average	Maximum	Median	Mode	Correlation	RMS
1	Training	6.8783	-0.0006	25.2227	-0.1103	-1.6127	1.0000	6.8766
1	Testing	5.7397	0.2785	30.0599	-0.0428	-2.3454	1.0000	5.7432
2	Training	12.7137	0.0000	95.5830	-0.4548	-2.0140	1.0000	12.7105
2	Testing	10.9267	-0.2080	52.3120	-0.5078	-31.4170	1.0000	10.9224
3	Training	18.3979	-0.0308	90.9622	-1.6689	-3.6689	1.0000	18.3933
3	Testing	17.4917	-1.8680	63.7925	-5.6689	-7.6689	1.0000	17.5812
4	Training	6.9312	0.0282	25.8888	-0.1814	-2.8192	1.0000	6.9296
4	Testing	6.4726	0.3032	30.8219	-0.2951	-4.4821	1.0000	6.4761

All the parameters of network performance above were calculated in terms of the error values.

The error value is defined as the difference between the expected output, and the calculated output by ANN.

Error = expected output - calculated output by ANN

$e = T - Y$

where:

e is the error scalar

T is the column target vector

Y is a column vector, the output of the transfer functions

$$y_h = f\,[W * P + B]$$

Std Dev is the standard deviation of e [22]. It is a measure of the variability or dispersion of a data set. A low standard deviation indicates that the data points tend to be very close to the same value (the mean); while high standard deviation indicates that the data are "spread out" over a large range of values.

$$\delta = \sqrt{\frac{\sum_{i=1}^{n}(e_i - \bar{e})^2}{n}}$$

Average is the average of e.

Maximum is the largest element in e.

Mode is the most frequent value in e.

The correlation coefficient [22] indicates the strength and direction of a linear relationship between variables.

$$R(i,j) = \frac{C(i,j)}{\sqrt{C(i,i)C(j,j)}}$$

where C is the covariance matrix of e [23]

RMS is the root mean square error [24]. It is a statistical measure of the magnitude of a varying quantity.

$$e_{rms} = \sqrt{\frac{\sum_{i=0}^{n} {e_i}^2}{n}}$$

The training parameters match the testing parameters among all the networks. The correlation coefficient is a statistical measure of how well the predictions agree with the targets. It can be seen that all the networks show a good correlation between the target and network output

As shown in Table 4-2, network 1 shows the lowest RMS error, lowest standard deviation of error, lowest maximum of error and lowest mode of error. Network 4 and network 1 are very similar. Although network 2 shows the lowest average of error, but it has higher standard deviation of error, higher maximum of error, higher mode of error and higher RMS error than network 1 and network 4. Although network 2 did a good job in terms of curve fitting, it lost the accuracy due to the large target excursion. Overall, network 1 provides the best results in the network performance. Network 4 provides the second best results, network 2 provides the third best results, and network 3 is the worst.

4.4 Comparison of Training Performance between each Network

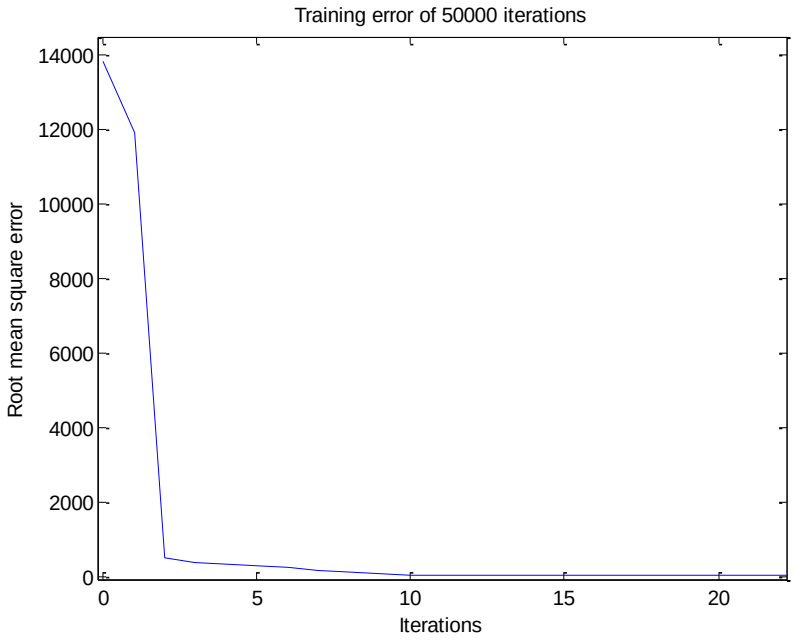

Figure 4-17: Training performance of Network 1

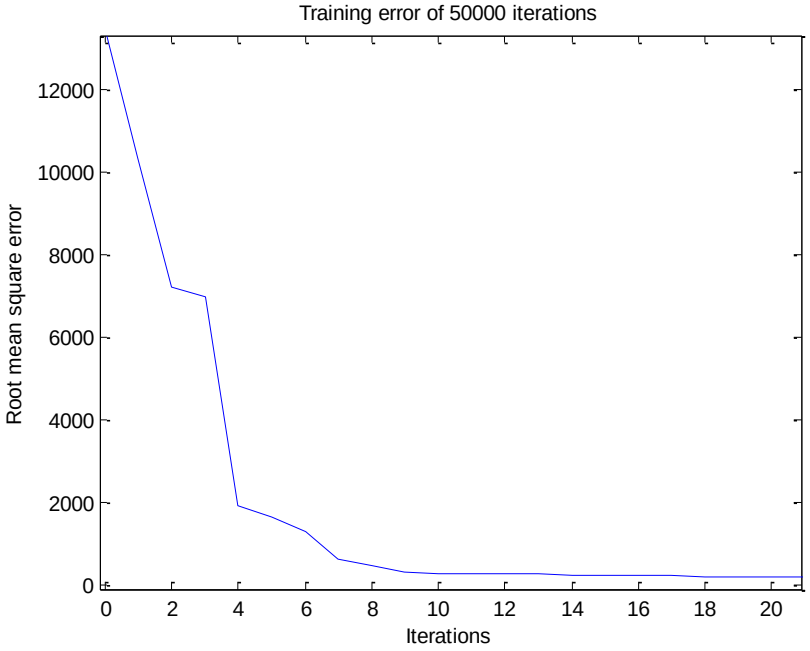

Figure 4-18: Training performance of Network 2

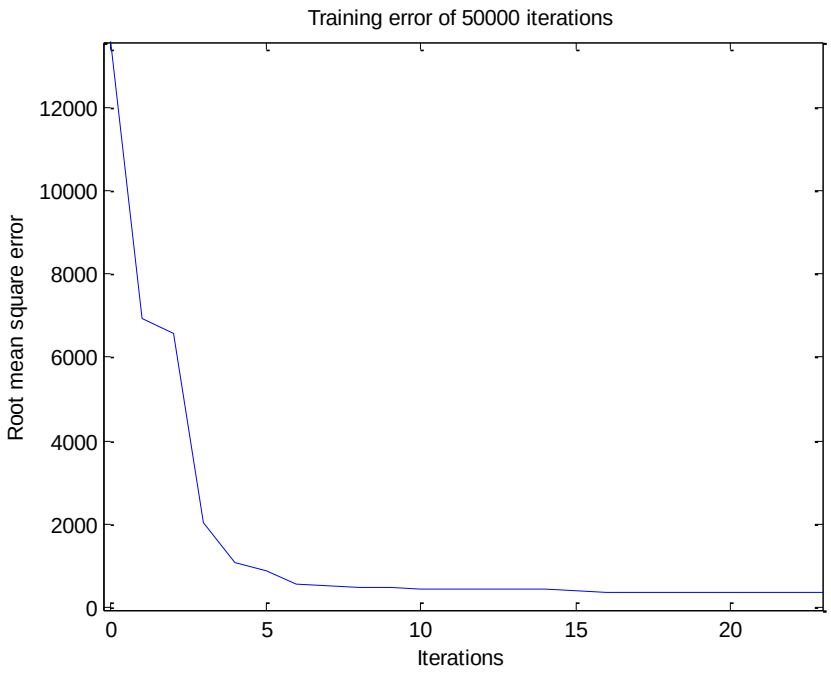

Figure 4-19: Training performance of Network 3

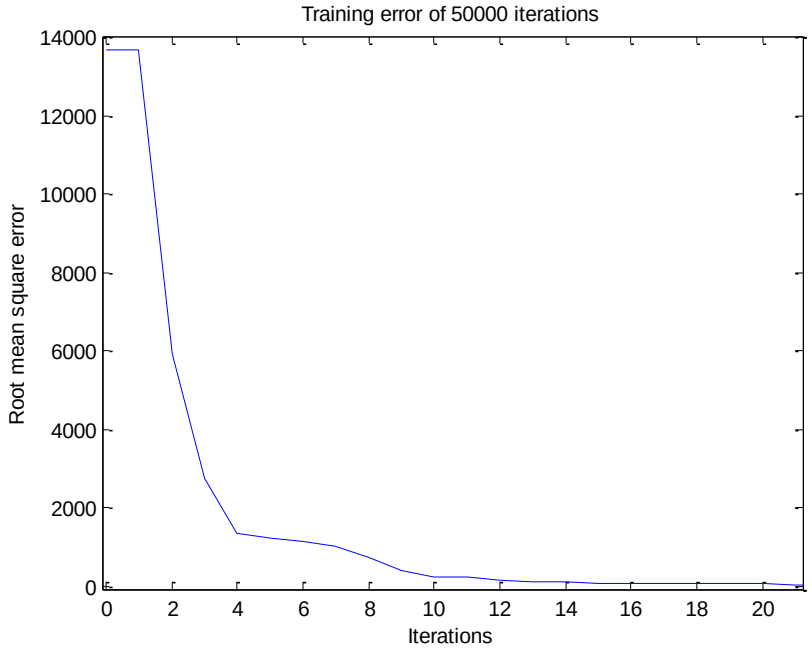

Figure 4-20: Training performance of Network 4

The training performance in Figure 4-17 shows that the training error of network 1 is reduced to the desired accuracy after 500000 iterations. The training performance in Figure 4-18 shows that the training error of network 2 only marginally converges after 50000 iterations. This may be caused by the large target excursion. The training performance in Figure 4-19 shows that the training error of network 3 only marginally converges after 50000 iterations. This may be caused by under-fitting. The training performance in Figure 4-20 shows that the training error of network 4 is reduced to the desired accuracy after 500000 iterations. The simulation results show that the predicted power consumptions of network 1 and network 4 are able to match the actual power consumptions with acceptable accuracy, but the network 2 and network 3 fail.

Chapter 5: Conclusions and Future Works

System modeling using artificial neural networks is a generic technique for mapping the relationship between inputs and outputs; it requires less expertise and experimentation than traditional modeling of non-linear multivariate systems. Once a neural network model is developed, engineers can easily apply it to predict and evaluate a particular building energy performance.

The results in this research show that multi-layer feed-forward artificial neural networks can provide reasonable accuracy for building power predictions. Adding too many hidden layers or neurons can decrease the accuracy because of over-fitting.

For future work, additional factors such as data noise elimination need to be addressed Data noise caused by errors generated from manual data measurements can be eliminated by installing continuous automatic measurement devices.

Last but not least, this research can go further by developing next generation artificial neural networks for predictive applications. It can be done by combining artificial neural networks with other algorithms, such as Kalman filters or regression methods.

Reference

[1] Wikimedia, “Neural network example,” http://commons.wikimedia.org/wiki/File:Neural_network_example.svg, May 2009.

[2] Ansett M, Kreider JF., “Application of neural networking models to predict energy use,” ASHRAE Transactions: Research 99(1):505–517, 1993.

[3] Curtiss PS., ‘Examples of neural networks used for building system control and energy management,” ASHRAE Transactions: Symposia BN 97-16-1:909–913, 1996.

[4] Cohen DA, Krarti M., “A neural network modeling approach applied to energy conservation retrofits,” Proceedings of Fourth International Conference on Building Simulation, Madison, WI, 423–430, 1995.

[5] Haberl JS, Thamilseran S., “Predicting hourly building energy use: the great energy predictor shootout II: measuring retrofit savings overview and discussion of results,” ASHRAE Transactions 102(Pt. 2):419–435, 1996.

[6] Breekweg MRB, Gruber P, Ahmed O., “Development of generalized neural network model to detect faults in building energy performance,” part I, part II. ASHRAE Transactions: Research 4372:61–93, 2000.

[7] Kreider JF, Claridge DE, Curtiss P, Haberl JS, Krarti M., “Building energy use prediction and system identification using recurrent networks,” transactions of the ASME. Journal of Solar Energy Engineering 117:161–166, 1995.

[8] Curtiss PS., “Examples of neural networks used for building system control and energy management,” ASHRAE Transactions: Symposia BN 97-16-1:909–913, 1996.

[9] Chonan Y, Nishida K, Matsumoto T., "Great energy predictor shootout II: a Bayesian nonlinear regression with multiple hyper-parameters," ASHRAE Transactions: Symposia SA-96-3-1:405-411, 1996.

[10] Jang K-J, Bartlett EB, Nelson RM., "Measuring retrofit energy savings using auto-associative networks," ASHRAE Transactions: Symposia SA-96-3-1:412–418, 1996.

[11] D. Datta, S.A. Tassou and D. Marriott, "Application of neural networks for the prediction of the energy consumption in a supermarket," In: Proc. CLIMA 2000 Conf., Brussels, Belgium, p. 98, 1997.

[12] L.Frosini and G. Petrecca, "Neural networks for energy flows prediction in facility systems," 1999 IEEE Midnight-Sun Workshop on Soft Computing Methods in Industrial Applications, Kuusamo, Finland, June 16-18, 1999.

[13] Melek Yalcintas1 and Sedat Akkurt, "Artificial neural networks applications in building energy predictions and a case study for tropical climates," INTERNATIONAL JOURNAL OF ENERGY RESEARCH, Int. J. Energy Res. 2005; 29:891–901, Wiley InterScience, 24 February 2005.

[14] John Seng, "Advanced Topics in Computer Engineering," CPE 482 Lecture Notes, California Polytechnic State University, San Louis Obispo, CA, Spring 2007

[15] Wikipedia, "Nearest-neighbor interpolation," http://en.wikipedia.org/wiki/Nearest-neighbor_interpolation, May 2009.

[16] No author, "How many hidden units should I use?" http://www.faqs.org/faqs/ai-faq/neural-nets/part3/section-9.html, May 2009.

[17] Geman, S., Bienenstock, E. and Doursat, R, "Neural Networks and the Bias/Variance Dilemma," Neural Computation, 4, 1-58, 1992.

[18] J. Sjoberg, H. Hjalmerson and L. Ljung, "Neural Networks in System Identification," 10th IFAC symposium on SYSID, Copenhagen, 1994.

[19] The MathWorks, Inc. “transfer function,” documentation search results of matlab help file, May 2009.

[20] The MathWorks, Inc. “Levenberg-Marquardt,” documentation search results of matlab help file, May 2009.

[21] The MathWorks, Inc. “learninglm,” documentation search results of matlab help file, May 2009.

[22] Wikipedia, “standard deviation,” http://en.wikipedia.org/wiki/Standard_deviation, May 2009.

[22] Wikipedia, “Correlation,” http://en.wikipedia.org/wiki/Correlation, May 2009.

[23] The MathWorks, Inc. “corrcoef,” documentation search results of matlab help file, May 2009.

[24] Wikipedia, “root mean square,” http://en.wikipedia.org/wiki/Root_mean_square, May 2009.

[25] DOE2, “Equest ...the QUick Energy Simulation Tool”, 3/9/2007, DOE2.

[26] James A. Freeman and David M. Skapura, “Neural Networks Algorithms, Applications, and Programming Techniques,” Addison-Wesley, 1992.

[27] Mohamad H. Hassoun, “Fundamentals of ARTIFICAL NEURAL NETWORKS,” The MIT Press, 1995.

[28] Jay L. Devore, “Probability and Statistics for Engineering and the Sciences,” Duxbury, 1999.

[29] Stephen J. Chapman, "MATLAB Programming for Engineers," Brooks/Cole, 2000.

[30] Howard Demuth, Mark Beale, Martin Hagan, "Neural Network Toolbox User's Guide," The MathWorks, Inc., 2008.

[31] Wikipedia, "Residual sum of squares," http://en.wikipedia.org/wiki/Sum_of_squared_residuals, June 2009.

Appendix A: Source Code

```
clc;
clear all;                          % Clear workspace
% ---------- Read data from text file ------------------------------
fid = fopen('powerBld1c.txt');              % Open text file

  pow = textscan(fid, '%n %n %n %n %n %n','commentStyle','#');

fclose(fid);                        % Close the text file

fid = fopen('Weatherc.txt');              % Open text file

  We = textscan(fid, '%n %n %n %n %n %f %f %f %n %n %n %f','commentStyle','#');
  %weather = [W{3} W{1} W{2} W{4} W{5} W{6} W{7} W{8} W{9} W{10} W{11} W{12}];
  %year = W{3};
fclose(fid);

%[S,R]=size(pow);
%[U,V]=size(We);

%Power
%DateAndTime    Val, kW
month = pow{1};                     % Store the first column as month
date = pow{2};                      % Store the second column as date
year = pow{3};                      % Store the third column as year
hr  = pow{4};                       % Store the 4th column as hour
min = pow{5};                       % Store the 5th column as minute
power = pow{6};                     % Store the 6th column as power consumptions

%totMin = hr*60 + min;
totMin = (date-1)*1440+hr*60+min;
%totMin = month*43200+date*1440+hr*60+min;

%Weather
%Date   Time    Temperature(F)  Dew Point(F)    Pressure(in)    Wind Speed(mph) Wind Gust(mph)
Humidity(%) Rainfall Rate(Hourly)
month2 = We{1};                     % Store the first column as month
date2 = We{2};                      % Store the second column as date
year2 = We{3};                      % Store the third column as year
hr2   = We{4};                      % Store the 4th column as hour
min2  = We{5};                      % Store the 5th column as minute
temp = We{6};                       % Store the 6th column as Temperature
dewP = We{7};                       % Store the 7th column as Dew Point
pres = We{8};                       % Store the 8th column as Pressure
windSp = We{9};                     % Store the 9th column as Wind Speed
windGu = We{10};                    % Store the 10th column as Wind Gust
hum = We{11};                       % Store the 11th column as Humidity
rain = We{12};                      % Store the 12th column as Rainfall Rate
```

```
totMin2 = (date2-1)*1440+hr2*60+min2;

%Sampling
time = 0:15:43200;
%interpolation, choose 'nearest', 'linear', 'spline' or 'pchip'
method='nearest';
ps = interp1(totMin,power,time,method);
temps = interp1(totMin2,temp,time,method);
dewPs = interp1(totMin2,dewP,time,method);
press = interp1(totMin2,pres,time,method);
windSps = interp1(totMin2,windSp,time,method);
windGus = interp1(totMin2,windGu,time,method);
hums = interp1(totMin2,hum,time,method);
rains = interp1(totMin2,rain,time,method);
psp = circshift(ps,[0 1]);

%P=[time' ps']
%format short;
%Table = [time' temps' dewPs' press' windSps' windGus' hums' rains' psp' ps']
I = [temps' dewPs' press' windSps' windGus' hums' rains' psp'];
[S,R]=size(I);

%P=I((3:(S-1)),:)';
%T=ps(3:(S-1));
P=I((3:2002),:)';
PT=I((2003:(S-1)),:)';
T=ps(3:2002);
TT=ps(2003:(S-1));
```

```
getData6;

% New NN and training and simulation
net = newff(minmax(P),[9 1],{'tansig' 'purelin'});
net.trainParam.epochs = 50000;
net.trainParam.lr = 0.01;
[net,tr,E] = train(net,P,T);
%Y = sim(net,P);
disp('Strucure #1, 8-9-1 tansig, purelin');

% Change to a new NN structure
%net = newff(minmax(P),[10 10 1],{'tansig' 'tansig' 'purelin'});
%net.trainParam.epochs = 50000;
%net.trainParam.lr = 0.01;
%[net,tr,E] = train(net,P,T);
%disp('Strucure #2, 8-10-10-1 tansig tansig, purelin');

%net = newff(minmax(P),[10 7 1],{'tansig' 'tansig' 'purelin'});
%net.trainParam.epochs = 50000;
%net.trainParam.lr = 0.01;
%[net,tr,E] = train(net,P,T);
%disp('Strucure #3, 8-10-7-1 tansig tansig, purelin');

%net = newff(minmax(P),[10 5 1],{'tansig' 'tansig' 'purelin'});
%net.trainParam.epochs = 50000;
%net.trainParam.lr = 0.01;
%[net,tr,E] = train(net,P,T);
%disp('Strucure #4, 8-10-5-1 tansig tansig, purelin');

%net = newff(minmax(P),[10 10 10 1],{'tansig' 'tansig' 'tansig'
'purelin'});
%net.trainParam.epochs = 50000;
%net.trainParam.lr = 0.01;
%[net,tr,E] = train(net,P,T);
%disp('Strucure #5, 8-10-10-10-1 tansig, tansig, tansig, purelin');

%Y = sim(net,P);

Testing4;
%reg;
plotData6;
```

```
MAX=max(power);
perf=tr.perf;
pL=length(perf);
%perf(pL)

%Training error
Y = sim(net,P);
eP=T-Y;
%ePn=eP/norm(eP); %normalize
%errorTrain=mse(ePn);
errorTrain=sqrt(perf(pL));

%Testing error
Y2 = sim(net,PT);
ePT=TT-Y2;
%ePTn=ePT/norm(ePT); %normalize
errorTest=sqrt(mse(ePT));

%e=[ePn ePTn];
e=[eP ePT];
%e=[eP ePT]

%Statics of power consumptions
er=power;
maxPo=max(er);          %Largest elements in array
meanPo=mean(er);        %Average or mean value of array
medianPo=median(er);       %Median value of array
modePo=mode(er);         %Most frequent values in array
stdPo=std(er);          %Standard deviation
statPo=[stdPo meanPo,maxPo,medianPo,modePo];


%Network Performance

%Training error
er=eP;  %error set
corP=corrcoef(er);    %Correlation coefficients
%covP=cov(er);          %Covariance matrix
maxP=max(er);          %Largest elements in array
meanP=mean(er);        %Average or mean value of array
medianP=median(er);       %Median value of array
modeP=mode(er);         %Most frequent values in array
stdP=std(er);          %Standard deviation
%varP=var(er);             %Variance
%perfP=[errorTrain,corP,covP,maxP,meanP,medianP,modeP,stdP,varP];
perfP=[stdP,meanP,maxP,medianP,modeP,corP,errorTrain];

%Testing error
er=ePT;  %error set
corPT=corrcoef(er);    %Correlation coefficients
%covPT=cov(er);          %Covariance matrix
maxPT=max(er);          %Largest elements in array
meanPT=mean(er);         %Average or mean value of array
medianPT=median(er);       %Median value of array
```

```
modePT=mode(er);         %Most frequent values in array
stdPT=std(er);          %Standard deviation
%varPT=var(er);            %Variance
%perfPT=[errorTest,corPT,covPT,maxPT,meanPT,medianPT,modePT,stdPT,varPT
];
perfPT=[stdPT,meanPT,maxPT,medianPT,modePT,corPT,errorTest];

%Printout the error
disp('Statistics of power consumptions');
disp('Std Mean Maximum Median Mode');
statPo
disp('Network Performance of training');
disp('Std Mean Maximum Median Mode Correlation RMS');
perfP
disp('Network Performance of testing');
disp('Std Mean Maximum Median Mode Correlation RMS');
perfPT
```

```
training=3:2002;
testing=2003:(S-1);
timeP=time(training); %time for training
timePT=time(testing); %time for testing
timeE=time(3:(S-1));
psP=ps(training); %power value during training
psPT=ps(testing); %power value during testing

figure(1);
%plot(time,ps,'o',time,Y,'-');
plot(time,ps);
title('Power consumptions in November 2008');
xlabel('Time(minutes)');
ylabel('Power consumptions(kW)');

figure(2);
plot(timeP,psP,timeP,Y,':');
title('Predicted power consumptions in November 2008 during training');
xlabel('Time(minutes)');
ylabel('Power consumptions(kW)');
legend('Target value', 'Predicted value using ANNs',1);

figure(3);
plot(timePT,psPT,timePT,Y2,':');
title('Predicted power consumptions in November 2008 during testing');
xlabel('Time(minutes)');
ylabel('Power consumptions(kW)');
legend('Target value', 'Predicted value using ANNs',1);

figure(4);
plot(timeE,e,'m');
title('Error of predicted power consumptions in November 2008');
xlabel('Time(minutes)');
ylabel('Error(kW)');

%plotperf(tr); %default
%plotperf(tr,NaN,'',100) %plot the first 100 iterations
figure(5);
epoch=tr.epoch;
perf=tr.perf;
plot(epoch,perf);
xlabel('Iterations');
ylabel('Root mean square error');
title('Training error of 50000 iterations');
```

Appendix B: Data

1. Cal Poly Power Consumption

Cal Poly electrical data (In CD)

Cal Poly Power Consumption Data no 1 (In CD)

Cal Poly Power Consumption Data no 2 (In CD)

Campus Energy (In CD)

2. Weather data of San Luis Obispo

http://www.wunderground.com/weatherstation